PROCEEDINGS

55. IWK

Internationales Wissenschaftliches Kolloquium
International Scientific Colloquium

13 - 17 September 2010

Crossing Borders within the ABC

Automation,

Biomedical Engineering and

Computer Science

Faculty of
Computer Science and Automation

www.tu-ilmenau.de

TECHNISCHE UNIVERSITÄT
ILMENAU

Impressum
Published by

Publisher:	Rector of the Ilmenau University of Technology Univ.-Prof. Dr. rer. nat. habil. Dr. h. c. Prof. h. c. Peter Scharff
Editor:	Marketing Department (Phone: +49 3677 69-2520) Andrea Schneider (conferences@tu-ilmenau.de) Faculty of Computer Science and Automation (Phone: +49 3677 69-2860) Univ.-Prof. Dr.-Ing. habil. Jens Haueisen
Editorial Deadline:	20. August 2010
Implementation:	Ilmenau University of Technology Felix Böckelmann Philipp Schmidt

USB-Flash-Version.

Publishing House:	Verlag ISLE, Betriebsstätte des ISLE e.V. Werner-von-Siemens-Str. 16 98693 Ilmenau
Production:	CDA Datenträger Albrechts GmbH, 98529 Suhl/Albrechts
Order trough:	Marketing Department (+49 3677 69-2520) Andrea Schneider (conferences@tu-ilmenau.de)

ISBN: 978-3-938843-53-6 (USB-Flash Version)

Online-Version:

Publisher:	Universitätsbibliothek Ilmenau ilmedia Postfach 10 05 65 98684 Ilmenau

Home / Index:
http://www.db-thueringen.de/servlets/DocumentServlet?id=16739

DEVELOPMENT OF AN ALGORITHM FOR CONVOY TASKS FULFILLED BY UGV IN PARTIALLY UNPAVED TERRAIN USING MODERN SIMULATION TECHNIQUES

Björn Steurer, Thomas Kopfstedt

Diehl BGT Defence GmbH & Co. KG
Alte Nussdorfer Strasse 13, 88662 Überlingen, Germany
bjoern.steurer@diehl-bgt-defence.de / thomas.kopfstedt@diehl-bgt-defence.de

beginabstract This paper will present new highly complex simulation environments using the Microsoft Robotics Developer Studio with the simulation of cameras in combination with 3D Lasers and simulated self localization algorithms as testbed for autonomous behaviors.
This simulation environment allows the increase of the development process and gives the chance to compare the capabilities of different algorithms under exactly the same conditions in large simulated environments considerating the physical effects and the latencies of the simulated sensors including error models.
We will present how this simulation environment is built using independent services and how this allowed us to adapt the simulation environment for different robotic systems running a Diehl BGT Defence. As an example we will present for one of our autonomous systems the development and transfer of an algorithm for object tracking and following for a convoy task from the simulation to the real system including experimental data.

Index Terms— Simulation Environment, Leader Following, CCD camera model, SLAM

1. INTRODUCTION

In today's world the delivery of all kinds of goods is often realized by the use of trucks that are driving in temporary convoys like on highways or specific created convoys. For example a convoy that is used for delivering relief supplies in cases of disasters. Until now all vehicles require a driver and these drivers are human beings who have to keep their resting times to avoid accidents.
In publications like [4], [7], [1] and [3] it has been shown that in human made environments it is possible to have autonomous vehicles that are following a leading vehicle. In urban environments with paved streets, on highways, or within buildings the terrain itself is not considered, because the focus is on keeping the convoy on the street and to react in the right manner on traffic signs and other traffic. In this context only legal issues are the blocking point as in case an autonomous vehicle is involved in a traffic accident the question of responsibility is not finally answered. As a consequence the car manufacturers are concentrating on supporting functionalities like line control, distance control, emergency breaking etc.
In outdoor environments and when there is danger for human beings, like after natural disasters or in dangerous regions of the world, this focus can change depending on the specific requirements and the specific risk for human lives. In areas that suffer from political instability or poverty, like some regions in Africa and in Afghanistan, the infrastructure is usually less developed and therefore the tarmac streets have no clearly identifiable white or yellow lines and in many cases even no tarmac street. In these areas different capabilities are required. Detection of street signs and detection of lines on the street is no longer in the main focus. Instead it is important to be able to create maps of the local surrounding in real-time containing information about the gradient, the height, and the type of surface of this local environment. The information from this map is used to find a path that allows mainly following along the track of the leading vehicle. If required the vehicle is capable to modify its path. Possible reasons for this are that different types of vehicles are used or that due to the movement of the leading vehicle the terrain has slightly changed when the following vehicle has to pass it, something that can easily happen on muddy or sandy ground.
Algorithms for coordinated movements of groups of autonomous vehicles have been investigated for example in [9] but the focus here is not set on fulfillment of convoy tasks with the direct consideration of the terrain during the path planning. Other interesting approaches, already in outdoor environment, have been discussed in [2] for different types of formations and a comparison between them was made.
In this paper we will present how algorithms for convoy tasks on unpaved terrain can be developed and how simulation techniques can help to reduce the number of required tests and help to speed up the development process. Current limitations due to the complexity of

the required simulations and the available development tools will be discussed as these limitations have indirect and direct effect on the real-time factor and the limits for simulations.
In chapter 2 we will give a short overview of possible software environments and the different requirements for test platforms, operational systems and simulation environments that are running in research and development departments. In chapter 3 the simulation environment is described and in chapter 4 the simulated UGV capabilities are introduced. Based on these chapters simulation results are presented in chapter 5 before finally some information about performed test runs is given in chapter 6.

2. SOFTWARE ENVIRONMENT

2.1. Short Comparison of existing simulation development environments

There are currently several different tools for simulation and development of algorithms for UGVs available. Some of them, like MATLAB and its free derivates, are powerful for analysis and modeling of general control issues, while tools like Mathematica are focused on analytical solutions. Several of these tools have integrated or closely linked visualization tools, but a complete simulation of a robot requires more. There are sensors like CCD and IR cameras or laser range finders that need to be modeled, and often the physical effects regarding the movement of the vehicle platform have to be considered to have more or less realistic models from the own robot for the development of UGV algorithms.
For simulation there are currently two mayor tools available. One is Player Stage with the GAZEBO extension for 3D simulation and the other is Microsoft Robotics Developer Studio (MRDS) in its current version 2008 R3. Both tools have a large number of followers, of which Player Stage is widely used especially in university context. Both have an integrated physics engine that requires a lot of CPU and GPU power for complex simulations. A detailed documentation for both tools is available in the Internet.
While Player Stage is using C++ as main language the MRDS can be programmed in Visual Basic, C++ and C# with C# as the most recommended language. In other points these tools are more or less identical, as both can be used for multi-core programming using services, and both have an integrated communication support to allow communication between services in an effective and reliable manner.

2.2. Comparison of existing environments for robot control

Regarding the operating systems for robots several solutions are possible. For robots that will later become a product no operating system but the direct use of FPGA or DSPs is often the best solution as this allows a short boot time, low power consumption and it helps during qualification of the code. For development test systems this is different as the focus is lieing on flexibility and the capability to allow fast changes of hardware and software components. As a consequence a ruggedized PC system is often a good compromise. As operating system Windows, Linux and real-time operating systems like VxWorks are the standard alternatives. While real-time operating systems allow a very accurate execution of services they have their bottleneck in the number of available drivers for external sensors etc. This is different for Windows and Linux as drivers for almost all external sensors are directly available and, depending on the specific settings of the operating system, even loop times of 10ms for execution can be ensured with less than 10% variation.

2.3. Specific requirements from industrial point of view

Comparing the specific requirements for the development with the available tools on the market on the one hand and the partially different focus that industrial business implies, like the point that buying a working software component or using a existing driver is more efficient than developing something existing in a slightly different version on its own, on the other hand, we are using Microsoft Robotics Developer Studio for simulation of robots and a Linux operating system for the test UGVs. This selection has the disadvantage that software has to be maintained and developed for two ”worlds” (Windows and Linux), but on the other hand this helps to develop software that is highly stable, modular, and well structured because operating system specific configurations are not allowed and the source code is tested with different compilers and can be checked on different platforms.

3. SIMULATION ENVIRONMENT

The simulation environment is created as a modular system with several independent services that are designed for fulfillment of specific tasks. This allows the reuse of services for different simulations of UGVs without modifications. Moreover, several developers are able to work at the same time on the overall simulation. Therefore, the possibility is given to have alternatives of different autonomous behaviors, sensor models, or robot models. Before the software is transferred into a robot the probably best performing combination of algorithms can already be selected by comparing their specific capabilities and their influence in simulation.

Fig. 1. Simulated environment: visual layer

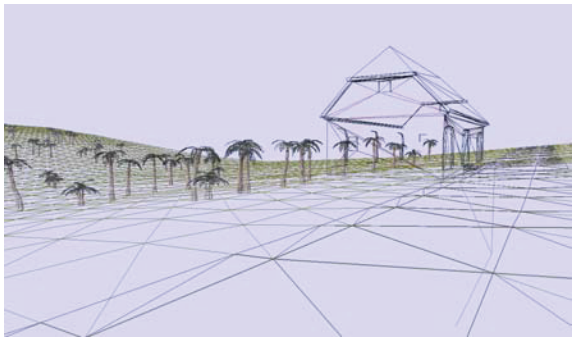

Fig. 2. Simulated environment: wire frame layer

3.1. Service architecture

To allow a flexible structure the following services have been realized for the simulation of the UGV with follower capabilities and terrain consideration:

- ⋆ simulation environment
- ⋆ dynamic objects (civilians / enemies)
- ⋆ CCD camera model
- ⋆ Laser range finder model
- ⋆ INS and GPS model
- ⋆ simulated communication
- ⋆ UGV platform model
- ⋆ UGV platform control (*)
- ⋆ image analysis (*)
- ⋆ object tracking (*)
- ⋆ self localization (*)
- ⋆ local map creation (*)
- ⋆ local path planning (*)
- ⋆ path following (*)

The services that are marked with (*) can be transferred to the test UGV while the others are designed for use in the simulation environment only. This distinction is relevant as software services that will be used on the test UGV must be real-time capable while for the software services that are only used in simulation this is not necessarily required and sometimes even not possible, as for example simulating a high speed rotating laser device like a Velodyne HDL 64 requires the calculation of more than 1.3 million simulated laser beams using raytracing, which exceeds the CPU capacity of many PCs available today for calculation in real-time.

3.2. Simulated world model

As sensors like CCD cameras and laser range finders are simulated it is required to have a more or less realistic appearance of objects in the simulation environment. The problem is, on the one side, that high polygon models of all objects inside the simulation would slow down the simulation and, on the other side, low polygon models do not deliver accurate data for the

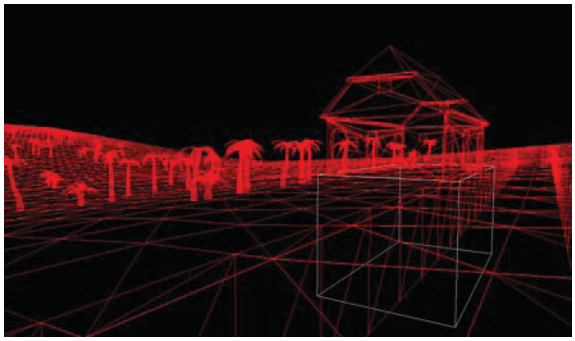

Fig. 3. Simulated environment: physics view layer

simulated laser range finder and the simulated CCD camera. Our solution is to use low polygon models with textures containing the additional detail information for the CCD camera image and to add a region depending jitter of several centimeters to the laser range finder data to allow a partial simulation of effects that also occur in reality when objects and environments are scanned by laser beams.
The simulation environment allows different views that are connected to the different types of sensors, and that are gathering signal data out of this simulated world. In Fig. 1 the data layer for the CCD camera is shown while in Fig. 2 the corresponding wire frame model, which can be used for the laser range finder beams, and in Fig. 3 the physics model is shown. The latter could alternatively be used for the laser range finder beams and it is used for the collision tests and the simulation of the robot platform behavior towards the ground.

3.3. Time used in simulation

As the simulation contains, beside the algorithms that are later running in the test UGV, several software services for the simulation of sensor data and the physical behavior of the UGV with a high level of detail, the simulation is even on an up-to-date multi-core machine with a high-end graphics card and an integrated hardware physics chip on the GPU not performing in

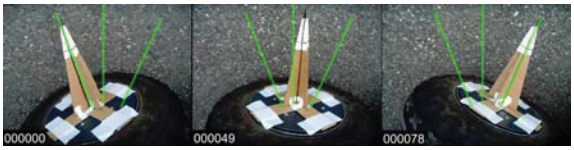

Fig. 4. Identification of the speed of changing the steering angle

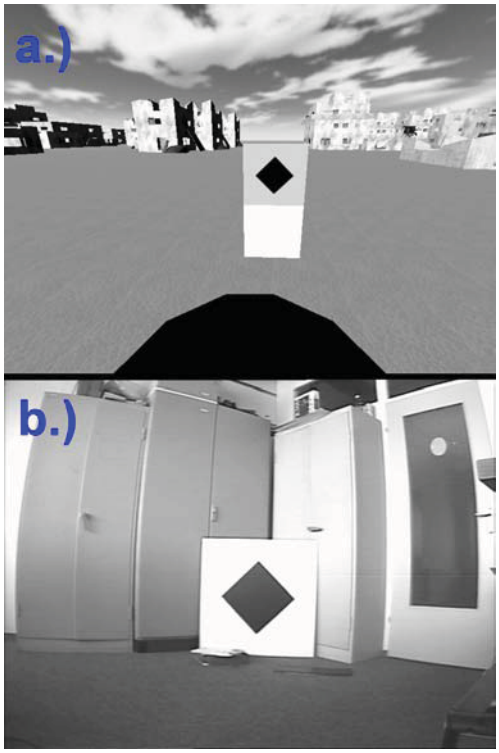

Fig. 5. Comparision of CCD camera a.) in simulation b.) from test UGV

real-time. However, the simulation allows an easier development and test of algorithms as identical mission can be performed several times and failures in the UGV software are not leading to hardware damage. A drawback, on the other side, is that effects that can be detected after ten minutes on a trial require sometimes simulations of up to one hour depending on the setting of the level of detail of the simulated sensors and the size of the simulated environment.

4. SIMULATED ROBOT BEHAVIORS

The simulated UGV is able to detect with its CCD camera a specific marker on the leading vehicle and to follow this marker by calculating the relative position of the leading vehicle towards the current own position. In addition to this the data from the laser range finder device is used for the creation of a digital map of the local surrounding. The UGV uses the algorithm published

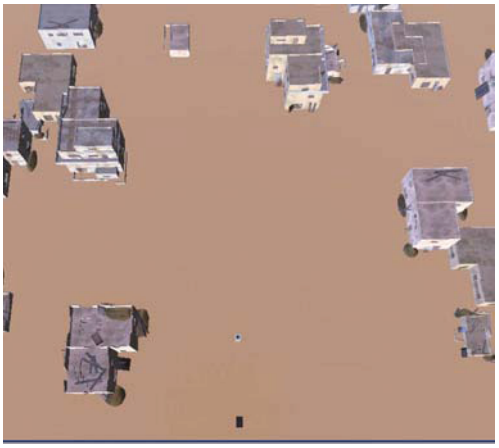

Fig. 6. View on the scene used for the simulated 2D laser

in [6] to consider the terrain while trying to follow the track of the leading vehicle. An alternative approach regarding the follow the leader task can be found in [5].

The capability of comparison of simulation results depends on the correspondence with real sensors that are used, and on the level of detail of the UGV test vehicle. To achieve a good model of the UGV platform the dynamic behavior of the UGV has been identified as shown for example in Fig. 4. In addition the simulated sensor data has to be compared with sensor data from sensors on the test UGV. An example for the CCD camera is shown in Fig. 5 which shows that the rhombic marker placed in the middle of the camera view is clearly visible in both cases, but lightening effects regarding reflection in the white parts of the gray pictures are different. This is something that needs to be considered and that can be partially solved by using two different parameter settings, one for the simulated camera and one for the camera on the test UGV.

As laser range finder data always requires interpretation for visualization in Fig. 7 only the simulated data of a 2D laser range finder of the scene from Fig. 6 is shown here. The difficulty with the simulation of laser beams is that in reality the laser beam has a width that is increasing with the distance. As a consequence small objects in larger distance have a larger jitter on the laser beam values and they are often not directly measured, while in simulation using raytracing the collision positions of the beams are considered as single points. Reflection intensity information is not simulated in our model.

5. SIMULATION RESULTS

Using the simulation model introduced in chapter 3 the algorithms for the tasks described in chapter 4 have been developed. Several tests have been performed in

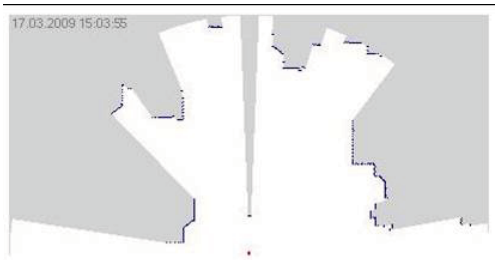

Fig. 7. Simulated 2D laser data

the simulation environment in order to verify the local map creation capabilities and to develop the algorithm for the object tracking and the task to follow the leader with consideration of the terrain surface.

As shown in Fig 8 the map creation is working well in the simulation, here visualized using Matlab, as inside the simulation the map is only used but no visualization is integrated. It can be seen that, depending on the gradient of the surface and the height of obstacles, the view in some direction is rather limited while in other directions the view is much better. This is something that has to be considered when local maps are used, because the map displayed only shows the information of the last laser scanner update. If more information is required a mapping of different maps using the self-localization information is required. With this, normally more detailed maps can be reached but depending on the quality of the self localization sometimes damaged maps are the result. For the task to follow the leader simulation runs have shown that mapping with old maps is normally not required, because the area of interest is a small region between the leading vehicle and the UGV. The advantage of this is that, even in cases when the self-localization is delivering disturbed data due to no GPS signal over a long period, the following itself remains stable. The capabilities of the task to follow the leader algorithm are shown in Fig. 9 where on a 100x100m ground the UGV has to follow a leading vehicle that is turning repeatedly. The presented data here shows only the tracks and not the environment that has to be considered additionally as the focus is here in the accuracy of the following along the desired trajectory and therefore the environment is kept flat without any obstacles. As shown here the UGV is able to follow accurately and the path planner has a good model of the UGV. The desired and the moved trajectory are only differing slightly.

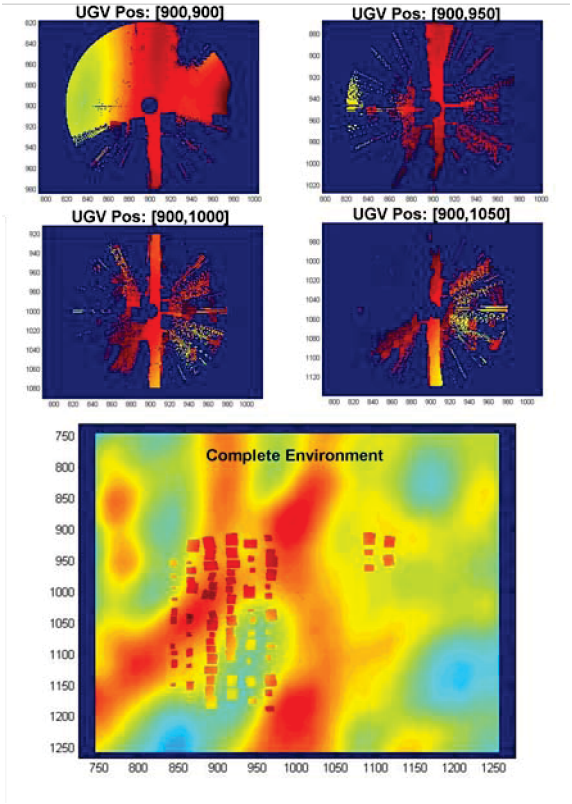

Fig. 8. Some local maps based on the laser scanner data

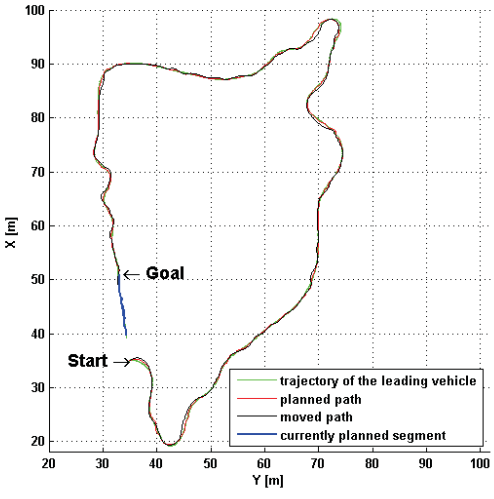

Fig. 9. Accuracy test of the leader following performed in simulation

6. EXPERIMENTAL RESULTS

The experimental test has been performed on the company ground with a human being carrying the track-

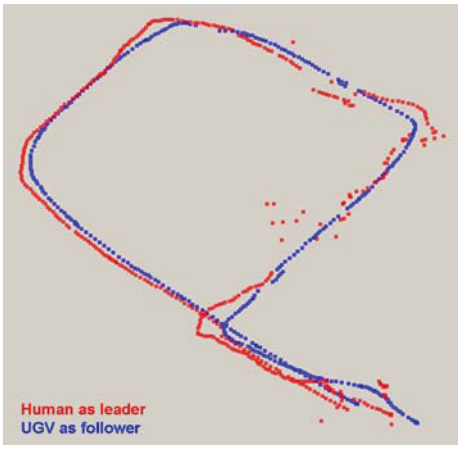

Fig. 10. UGV following a human being carrying the tracking object

ing object and a GPS receiver, and the test UGV with a DGPS receiver and a running self-localization algorithm. As a consequence the track from the human being is sometimes disturbed when due to the environment the GPS signal was not available or only with a low quality while the track from the UGV is collected with a higher quality as shown in Fig. 10. It can be seen in this figure that after the start the UGV is at first following with a lateral offset before it follows mainly in line of the track from the tracking object. In the corners the UGV is leaving the track generated by the human being with the tracking object as the UGV has a different turning radius and it additionally has to ensure that the view on the tracking object is maintained, because in this test the tracking was performed exclusively based on the estimated position of the tracking object using the CCD camera as relative distance and orientation to the calculated position of the UGV. This shows that the detection algorithm of the tracking object, the self-localization, the path planning and following as well as the robot control are working well together with a laser device that has been used for the creation of the map.

7. CONCLUSION AND OUTLOOK

As shown in chapter 6 and chapter 5 the developed algorithms are working well in the simulation and also in experimental settings. This shows that it is possible to develop algorithms directly in simulation environments with simulated sensors and to transfer the resulting algorithms and software components into a test platform with only some parameter setting modifications. Regarding low level vehicle control this concept is currently not possible as our tests have shown that the friction effects and vibration of the UGV due to unpaved terrain are not been handed well by the simulation effect with the required level of detail. The capability shown in this paper allows continuous autonomous following a leading vehicle or a human being. This is possible even in unpaved terrain where the capabilities of the leading vehicle or leading human can differ from the capabilities of the following UGV. Therefore an algorithm for the terrain-based following has been introduced and explained which allows the modification of the track of the leading vehicle based on the specific requirements of the follower UGV.

8. REFERENCES

[1] G. Antonelli, S. Chiaverini, "Kinematic Control of Platoons of Autonomous Vehicles", *IEEE Transactions on Robotics*, Vol. 22, No. 6, pp. 1285-1292, December 2006

[2] T. Balch and R. C. Arkin, "Behavior-based Formation Control for Multi-robot Teams", *IEEE Transactions on Robotics and Automation*, Vol. 16, No. 6, pp.926-939, December 1998

[3] C. Canudas-de-Wit, A.D. NDoudi-Likoho, "Nonlinear control for a convoy-like vehicle", *Automatica*, Vol. 36, pp. 457-462, 2000

[4] J. Fredslund, M.J. Mataric, "A General Algorithm for Robot Formation Using Local Sensing and Minimal Communication", *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 5, pp. 837-846, October 2002

[5] J. Huang, S.M. Farritor, A. Qadi, S. Goddard, "Localization and Follow-the-Leader Control of a Heterogeneous Group of Mobile Robots", *IEEE/ASME Transactions on Mechatronics*, Vol. 11, No. 2, pp. 205-215, April 2006

[6] T. Kopfstedt, M. Restle, W. Grimm, "Terrain Optimized Nonholonmic Following of Vehicle Tracks", *7th IFAC Symposium on Intelligent Autonomous Vehicles*, September 6-8, 2010, Lecce, Italy

[7] J. Shao, G. Xie, L. Wang, "Leader-following formation control of multiple mobile vehicles", *IET Control Theory Application*, Vol. 1, No. 2, pp. 545-552, 2007

[8] H. Seraji, A. Howard, "Behavior-Based Robot Navigation on Challenging Terrain: A Fuzzy Logic Approach", *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 3, pp. 308-321, June 2002

[9] L.A. Weitz, J.E. Hurtado, "Decentralized Cooperative-Control Design for Multivehicle Formations", *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 4, pp. 970-979, July-August 2008